

10/19/00
C904 U.S. PTO

10-20-00 EK 538216795 US
A

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Docket No. AUS9-2000-0628-US1

Assistant Commissioner for Patents
Washington, D.C. 20231

JC759 U.S. PTO
09/692351
10/19/00

Sir:

Transmitted herewith for filing is the patent application of Inventor(s):

Thomas Anthony Gregg,
Gregory Francis Pfister,
Renato John Recio, and
Steven Mark Thurber

For: END NODE PARTITIONING USING LMC FOR A SYSTEM AREA NETWORK

Enclosed are also:

- ☒ 26 Pages of Specification including an Abstract
- ☒ 4 Pages of Claims
- ☒ 8 Sheet(s) of Drawings
- ☒ A Declaration and Power of Attorney (2 Declarations)
- ☒ Form PTO 1595 and assignment of the invention to IBM Corporation

CLAIMS AS FILED

FOR	Number Filed		Number Extra		Rate		Basic Fee (\$710)
Total Claims	23	-20 =	3	X	\$ 18	=	\$54.00
Independent Claims	3	-3 =	0	X	\$ 80	=	\$0.00
Multiple Dependent Claims	0			X	\$270	=	\$
Total Filing Fee							= \$764.00

☒ Please charge \$764.00 to IBM Corporation, Deposit Account No. 09-0447.

☒ The Commissioner is hereby authorized to charge payment of the following fees associated with the communication or credit any over payment to IBM Corporation, Deposit Account No. 09-0447. A duplicate copy of this sheet is enclosed.

☒ Any additional filing fees required under 37CFR § 1.16.

☒ Any patent application processing fees under 37CFR § 1.17.

Respectfully,

Mark E. McBurney
Mark E. McBurney
Reg. No. 33,114
Intellectual Property Law Dept.
IBM Corporation
11400 Burnet Road 4054
Austin, Texas 75758
Telephone: (512) 823-1003

Docket No. AUS9-2000-0628-US1

END NODE PARTITIONING USING LMC FOR A SYSTEM AREA NETWORK

CROSS REFERENCES TO RELATED APPLICATIONS

The present invention is related to applications entitled A System Area Network of End-to-End Context via Reliable Datagram Domains, serial no. _____, attorney docket no. AUS9-2000-0625-US1; Method and Apparatus for Pausing a Send Queue without Causing Sympathy Errors, serial no. _____, attorney docket no. AUS9-2000-0626-US1; End Node Partitioning using LMC for a System Area Network, serial no. _____, attorney docket no. AUS9-2000-0628-US1; Method and Apparatus for Dynamic Retention of System Area Network Management Information in Non-Volatile Store, serial no. _____, attorney docket no. AUS9-2000-0629-US1; Method and Apparatus for Retaining Network Security Settings Across Power Cycles, serial no. _____, attorney docket no. AUS9-2000-0630-US1; Method and Apparatus for Reporting Unauthorized Attempts to Access Nodes in a Network Computing System, serial no. _____, attorney docket no. AUS9-2000-0631-US1; Method and Apparatus for Reliably Choosing a Master Network Manager During Initialization of a Network Computing System, serial no. _____, attorney docket no. AUS9-2000-0632-US1; Method and Apparatus for Ensuring Scalable Mastership During Initialization of a System Area Network, serial no. _____, attorney docket no. AUS9-2000-0633-US1; and Method and Apparatus for Using a Service ID for the Equivalent of Port ID in a Network Computing System, serial no. _____, attorney docket no. AUS9-2000-0634-US1, all of which are filed even date

Docket No. AUS9-2000-0628-US1

hereof, assigned to the same assignee, and incorporated herein by reference.

BACKGROUND OF THE INVENTION

1. Technical Field:

The present invention relates generally to an improved data processing system, and in particular to a method and apparatus for identifying connections between data processing systems. Still more particularly, the present invention provides a method and apparatus for establishing connections with a process on a data processing system with partitioning of System Area Network (SAN) end nodes to accommodate Logical Partitioning (LPAR).

2. Description of Related Art:

In a System Area Network (SAN), the hardware provides a message passing mechanism which can be used for Input/Output devices (I/O) and interprocess communications between general computing nodes (IPC). Consumers access SAN message passing hardware by posting send/receive messages to send/receive work queues on a SAN channel adapter (CA). The send/receive work queues (WQ) are assigned to a consumer as a queue pair (QP). The messages can be sent over five different transport types: Reliable Connected (RC), Reliable datagram (RD), Unreliable Connected (UC), Unreliable Datagram (UD), and Raw Datagram (RawD). Consumers retrieve the results of these messages from a completion queue (CQ) through SAN send and receive work completions (WC). The source

Docket No. AUS9-2000-0628-US1

channel adapter takes care of segmenting outbound messages and sending them to the destination. The destination channel adapter takes care of reassembling inbound messages and placing them in the memory space designated by the destination's consumer. Two channel adapter types are present, a host channel adapter (HCA) and a target channel adapter (TCA). The host channel adapter is used by general purpose computing nodes to access the SAN fabric. Consumers use SAN verbs to access host channel adapter functions. The software that interprets verbs and directly accesses the channel adapter is known as the channel interface (CI).

A Local Identification (LID) refers to a short address used to identify a Channel Adapter (CA) port within a single subnet. A source LID (SLID) and a destination LID (DLID) are the source and destination LIDs used in a SAN communication packet header. The LID Mask Control (LMC) field in the SAN communication packet identifies the number of low order bits that can be used for routing the packet through different paths in the network. This means that a single CA port has up to 2^{LMC} LIDs assigned to it.

It is common for a system to be divided into Logical Partitions (LPARs) where various resources are allocated to a various user. For example, some systems can have various processors dedicated to one instance of the operating system and other processors in the same system allocated to another instance. The purpose of LPAR is so that if there is failure in the hardware or software, at most just the users in the same partition are affected and not the other partitions.

The problem occurs on how to route SAN packets to

the correct partitions in an efficient manner, when those partitions are sharing a SAN port.

SUMMARY OF THE INVENTION

The present invention provides a distributed computing system having end nodes, switches, routers, and links interconnecting these components. Each end nodes uses send and receive queue pairs to transmit and receives messages. The end nodes segment the message into packets and transmit the packets over the links. The switches and routers interconnects the end nodes and route the packets to the appropriate end node. The end nodes reassemble the packets into a message at the destination.

The present invention solves the problem of routing SAN packets to the correct partitions in an efficient manner, when those partitions are sharing a SAN port, by using the Local ID Mask Control (LMC) field to assign different LIDs to different partitions, and then using those low order bits to route the message to the appropriate partition.

BRIEF DESCRIPTION OF THE DRAWINGS

The novel features believed characteristic of the invention are set forth in the appended claims. The invention itself, however, as well as a preferred mode of use, further objectives and advantages thereof, will best be understood by reference to the following detailed description of an illustrative embodiment when read in conjunction with the accompanying drawings, wherein:

Figure 1 is a diagram of a network computing system is illustrated in accordance with a preferred embodiment of the present invention;

Figure 2 is a functional block diagram of a host processor node in accordance with a preferred embodiment of the present invention;

Figure 3 is a diagram of a host channel adapter in accordance with a preferred embodiment of the present invention;

Figure 4 is a diagram illustrating processing of work requests in accordance with a preferred embodiment of the present invention;

Figure 5 is an illustration of a data packet in accordance with a preferred embodiment of the present invention;

Figure 6 is a diagram of a communication over a portion of a SAN fabric;

Figure 7 depicts a schematic diagram illustrating the network addressing used in a distributed networking system in accordance with the present invention; and

Figure 8 depicts a schematic diagram illustrating

Docket No. AUS9-2000-0628-US1

the partition addressing using the LMC field in accordance with the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

The present invention provides a distributed computing system having end nodes, switches, routers, and links interconnecting these components. Each end node uses send and receive queue pairs to transmit and receives messages. The end nodes segment the message into packets and transmit the packets over the links. The switches and routers interconnects the end nodes and route the packets to the appropriate end node. The end nodes reassemble the packets into a message at the destination.

With reference now to the figures and in particular with reference to **Figure 1**, a diagram of a network computing system is illustrated in accordance with a preferred embodiment of the present invention. The network computing system represented in **Figure 1** takes the form of a system area network (SAN) **100** and is provided merely for illustrative purposes, and the embodiments of the present invention described below can be implemented on computer systems of numerous other types and configurations. For example, computer systems implementing the present invention can range from a small server with one processor and a few input/output (I/O) adapters to massively parallel supercomputer systems with hundreds or thousands of processors and thousands of I/O adapters. Furthermore, the present invention can be implemented in an infrastructure of remote computer systems connected by an internet or intranet.

SAN **100** is a high-bandwidth, low-latency network interconnecting nodes within the network computing

system. A node is any component attached to one or more links of a network. In the depicted example, SAN **100** includes nodes in the form of host processor node **102**, host processor node **104**, redundant array independent disk (RAID) subsystem node **106**, switch node **112**, switch node **114**, router node **117**, and I/O chassis node **108**. The nodes illustrated in **Figure 1** are for illustrative purposes only, as SAN **100** can connect any number and any type of independent processor nodes, and I/O adapter nodes. Any one of the nodes can function as an end node, which is herein defined to be a device that originates or finally consumes messages or frames in SAN **100**.

In one embodiment of the present invention, an error handling mechanism in distributed computer systems is present in which the error handling mechanism allows for reliable connection or reliable datagram communication between end nodes in distributed computing system, such as SAN **100**.

A message, as used herein, is an application-defined unit of data exchange, which is a primitive unit of communication between cooperating processes. A packet is one unit of data encapsulated by a networking protocol headers and/or trailer. The headers generally provide control and routing information for directing the frame through SAN. The trailer generally contains control and cyclic redundancy check (CRC) data for ensuring packets are not delivered with corrupted contents.

SAN **100** contains the communications and management infrastructure supporting both I/O and interprocessor communications (IPC) within a network computing system. The SAN **100** shown in **Figure 1** includes a switched

communications fabric, which allows many devices to concurrently transfer data with high-bandwidth and low latency in a secure, remotely managed environment. Endnodes can communicate over multiple ports and utilize multiple paths through the SAN fabric. The multiple ports and paths through the SAN shown in **Figure 1** can be employed for fault tolerance and increased bandwidth data transfers.

The SAN **100** in **Figure 1** includes switch **112**, switch **114**, switch **146**, and router **117**. A switch is a device that connects multiple links together and allows routing of packets from one link to another link within a subnet using a small header Destination Local Identifier (DLID) field. A router is a device that connects multiple subnets together and is capable of routing frames from one link in a first subnet to another link in a second subnet using a large header Destination Globally Unique Identifier (DGUID).

In one embodiment, a link is a full duplex channel between any two network fabric elements, such as endnodes, switches, or routers. Example suitable links include, but are not limited to, copper cables, optical cables, and printed circuit copper traces on backplanes and printed circuit boards.

For reliable service types, endnodes, such as host processor endnodes and I/O adapter endnodes, generate request packets and return acknowledgment packets. Switches and routers pass packets along, from the source to the destination. Except for the variant CRC trailer field which is updated at each stage in the network, switches pass the packets along unmodified. Routers update the variant CRC trailer field and modify other

fields in the header as the packet is routed.

In SAN **100** as illustrated in **Figure 1**, host processor node **102**, host processor node **104**, RAID I/O subsystem **106**, and I/O chassis **108** include at least one channel adapter (CA) to interface to SAN **100**. In one embodiment, each channel adapter is an endpoint that implements the channel adapter interface in sufficient detail to source or sink packets transmitted on SAN fabric **100**. Host processor node **102** contains channel adapters in the form of host channel adapter **118** and host channel adapter **120**. Host processor node **104** contains host channel adapter **122** and host channel adapter **124**. Host processor node **102** also includes central processing units **126-130** and a memory **132** interconnected by bus system **134**. Host processor node **104** similarly includes central processing units **136-140** and a memory **142** interconnected by a bus system **144**.

Host channel adapters **118** and **120** provide a connection to switch **112** while host channel adapters **122** and **124** provide a connection to switches **112** and **114**.

In one embodiment, a host channel adapter is implemented in hardware. In this implementation, the host channel adapter hardware offloads much of central processing unit and I/O adapter communication overhead. This hardware implementation of the host channel adapter also permits multiple concurrent communications over a switched network without the traditional overhead associated with communicating protocols. In one embodiment, the host channel adapters and SAN **100** in **Figure 1** provide the I/O and interprocessor communications (IPC) consumers of the network computing

Docket No. AUS9-2000-0628-US1

system with zero processor-copy data transfers without involving the operating system kernel process, and employs hardware to provide reliable, fault tolerant communications.

As indicated in **Figure 1**, router **117** is coupled to wide area network (WAN) and/or local area network (LAN) connections to other hosts or other routers.

The I/O chassis **108** in **Figure 1** includes a switch **146** and multiple I/O modules **148-156**. In these examples, the I/O modules take the form of adapter cards. Example adapter cards illustrated in **Figure 1** include a SCSI adapter card for I/O module **148**; an adapter card to fiber channel hub and fiber channel-arbitrated loop (FC-AL) devices for I/O module **152**; an ethernet adapter card for I/O module **150**; a graphics adapter card for I/O module **154**; and a video adapter card for I/O module **156**. Any known type of adapter card can be implemented. I/O adapters also include a switch in the I/O adapter backplane to couple the adapter cards to the SAN fabric. These modules contain target channel adapters **158-166**.

In this example, RAID subsystem node **106** in **Figure 1** includes a processor **168**, a memory **170**, a target channel adapter (TCA) **172**, and multiple redundant and/or striped storage disk unit **174**. Target channel adapter **172** can be a fully functional host channel adapter.

SAN **100** handles data communications for I/O and interprocessor communications. SAN **100** supports high-bandwidth and scalability required for I/O and also supports the extremely low latency and low CPU overhead required for interprocessor communications. User clients can bypass the operating system kernel process and

directly access network communication hardware, such as host channel adapters, which enable efficient message passing protocols. SAN **100** is suited to current computing models and is a building block for new forms of I/O and computer cluster communication. Further, SAN **100** in **Figure 1** allows I/O adapter nodes to communicate among themselves or communicate with any or all of the processor nodes in network computing system. With an I/O adapter attached to the SAN **100**, the resulting I/O adapter node has substantially the same communication capability as any host processor node in SAN **100**.

Turning next to **Figure 2**, a functional block diagram of a host processor node is depicted in accordance with a preferred embodiment of the present invention. Host processor node **200** is an example of a host processor node, such as host processor node **102** in **Figure 1**. In this example, host processor node **200** shown in **Figure 2** includes a set of consumers **202-208**, which are processes executing on host processor node **200**. Host processor node **200** also includes channel adapter **210** and channel adapter **212**. Channel adapter **210** contains ports **214** and **216** while channel adapter **212** contains ports **218** and **220**. Each port connects to a link. The ports can connect to one SAN subnet or multiple SAN subnets, such as SAN **100** in **Figure 1**. In these examples, the channel adapters take the form of host channel adapters.

Consumers **202-208** transfer messages to the SAN via the verbs interface **222** and message and data service **224**. A verbs interface is essentially an abstract description of the functionality of a host channel adapter. An operating system may expose some or all of the verb

Docket No. AUS9-2000-0628-US1

functionality through its programming interface. Basically, this interface defines the behavior of the host. Additionally, host processor node **200** includes a message and data service **224**, which is a higher level interface than the verb layer and is used to process messages and data received through channel adapter **210** and channel adapter **212**.

With reference now to **Figure 3**, a diagram of a host channel adapter is depicted in accordance with a preferred embodiment of the present invention. Host channel adapter **300** shown in **Figure 3** includes a set of queue pairs (QPs) **302-310**, which are used to transfer messages to the host channel adapter ports **312-316**. Buffering of data to host channel adapter ports **312-316** is channeled through virtual lanes (VL) **318-334** where each VL has its own flow control. Subnet manager configures channel adapters with the local addresses for each physical port, i.e., the port's LID. Subnet manager agent (SMA) **336** is the entity that communicates with the subnet manager for the purpose of configuring the channel adapter. Memory translation and protection (MTP) **338** is a mechanism that translates virtual addresses to physical addresses and to validate access rights. Direct memory access (DMA) **340** provides for direct memory access operations using memory **340** with respect to queue pairs **302-310**.

A single channel adapter, such as the host channel adapter **300** shown in **Figure 3**, can support thousands of queue pairs. By contrast, a target channel adapter in an I/O adapter typically supports a much smaller number of queue pairs.

Each queue pair consists of a send work queue (SWQ) and a receive work queue. The send work queue is used to send channel and memory semantic messages. The receive work queue receives channel semantic messages. A consumer calls an operating-system specific programming interface, which is herein referred to as verbs, to place work requests (WRs) onto a work queue.

With reference now to **Figure 4**, a diagram illustrating processing of work requests is depicted in accordance with a preferred embodiment of the present invention. In **Figure 4**, a receive work queue **400**, send work queue **402**, and completion queue **404** are present for processing requests from and for consumer **406**. These requests from consumer **402** are eventually sent to hardware **408**. In this example, consumer **406** generates work requests **410** and **412** and receives work completion **414**. As shown in **Figure 4**, work requests placed onto a work queue are referred to as work queue elements (WQEs).

Send work queue **402** contains work queue elements (WQEs) **422-428**, describing data to be transmitted on the SAN fabric. Receive work queue **400** contains work queue elements (WQEs) **416-420**, describing where to place incoming channel semantic data from the SAN fabric. A work queue element is processed by hardware **408** in the host channel adapter.

The verbs also provide a mechanism for retrieving completed work from completion queue **404**. As shown in **Figure 4**, completion queue **404** contains completion queue elements (CQEs) **430-436**. Completion queue elements contain information about previously completed work queue elements. Completion queue **404** is used to create a single

point of completion notification for multiple queue pairs. A completion queue element is a data structure on a completion queue. This element describes a completed work queue element. The completion queue element contains sufficient information to determine the queue pair and specific work queue element that completed. A completion queue context is a block of information that contains pointers to, length, and other information needed to manage the individual completion queues.

Example work requests supported for the send work queue **402** shown in **Figure 4** are as follows. A send work request is a channel semantic operation to push a set of local data segments to the data segments referenced by a remote node's receive work queue element. For example, work queue element **428** contains references to data segment 4 **438**, data segment 5 **440**, and data segment 6 **442**. Each of the send work request's data segments contains a virtually contiguous memory region. The virtual addresses used to reference the local data segments are in the address context of the process that created the local queue pair.

A remote direct memory access (RDMA) read work request provides a memory semantic operation to read a virtually contiguous memory space on a remote node. A memory space can either be a portion of a memory region or portion of a memory window. A memory region references a previously registered set of virtually contiguous memory addresses defined by a virtual address and length. A memory window references a set of virtually contiguous memory addresses which have been bound to a previously registered region.

The RDMA Read work request reads a virtually

Docket No. AUS9-2000-0628-US1

contiguous memory space on a remote end node and writes the data to a virtually contiguous local memory space. Similar to the send work request, virtual addresses used by the RDMA Read work queue element to reference the local data segments are in the address context of the process that created the local queue pair. For example, work queue element **416** in receive work queue **400** references data segment 1 **444**, data segment 2 **446**, and data segment **448**. The remote virtual addresses are in the address context of the process owning the remote queue pair targeted by the RDMA Read work queue element.

A RDMA Write work queue element provides a memory semantic operation to write a virtually contiguous memory space on a remote node. The RDMA Write work queue element contains a scatter list of local virtually contiguous memory spaces and the virtual address of the remote memory space into which the local memory spaces are written.

A RDMA FetchOp work queue element provides a memory semantic operation to perform an atomic operation on a remote word. The RDMA FetchOp work queue element is a combined RDMA Read, Modify, and RDMA Write operation. The RDMA FetchOp work queue element can support several read-modify-write operations, such as Compare and Swap if equal.

A bind (unbind) remote access key (R_Key) work queue element provides a command to the host channel adapter hardware to modify (destroy) a memory window by associating (disassociating) the memory window to a memory region. The R_Key is part of each RDMA access and is used to validate that the remote process has permitted access to the buffer.

In one embodiment, receive work queue **400** shown in **Figure 4** only supports one type of work queue element, which is referred to as a receive work queue element. The receive work queue element provides a channel semantic operation describing a local memory space into which incoming send messages are written. The receive work queue element includes a scatter list describing several virtually contiguous memory spaces. An incoming send message is written to these memory spaces. The virtual addresses are in the address context of the process that created the local queue pair.

For interprocessor communications, a user-mode software process transfers data through queue pairs directly from where the buffer resides in memory. In one embodiment, the transfer through the queue pairs bypasses the operating system and consumes few host instruction cycles. Queue pairs permit zero processor-copy data transfer with no operating system kernel involvement. The zero processor-copy data transfer provides for efficient support of high-bandwidth and low-latency communication.

When a queue pair is created, the queue pair is set to provide a selected type of transport service. In one embodiment, a network computing system implementing the present invention supports four types of transport services.

Reliable and Unreliable connected services associate a local queue pair with one and only one remote queue pair. Connected services require a process to create a queue pair for each process which is to communicate with over the SAN fabric. Thus, if each of N host processor nodes contain P processes, and all P processes on each

Docket No. AUS9-2000-0628-US1

node wish to communicate with all the processes on all the other nodes, each host processor node requires $P^2 \times (N - 1)$ queue pairs. Moreover, a process can connect a queue pair to another queue pair on the same host channel adapter.

Reliable datagram service associates a local end-end (EE) context with one and only one remote end-end context. The reliable datagram service permits a client process of one queue pair to communicate with any other queue pair on any other remote node. At a receive work queue, the reliable datagram service permits incoming messages from any send work queue on any other remote node. The reliable datagram service greatly improves scalability because the reliable datagram service is connectionless. Therefore, an end node with a fixed number of queue pairs can communicate with far more processes and endnodes with a reliable datagram service than with a reliable connection transport service. For example, if each of N host processor nodes contain P processes, and all P processes on each node wish to communicate with all the processes on all the other nodes, the reliable connection service requires $P^2 \times (N - 1)$ queue pairs on each node. By comparison, the connectionless reliable datagram service only requires P queue pairs + $(N - 1)$ EE contexts on each node for exactly the same communications.

The unreliable datagram service is connectionless. The unreliable datagram service is employed by management applications to discover and integrate new switches, routers, and endnodes into a given network computing system. The unreliable datagram service does not provide the reliability guarantees of the reliable connection

Docket No. AUS9-2000-0628-US1

service and the reliable datagram service. The unreliable datagram service accordingly operates with less state information maintained at each end node.

Turning next to **Figure 5**, an illustration of a data packet is depicted in accordance with a preferred embodiment of the present invention. Message data **500** contains data segment 1 **502**, data segment 2 **504**, and data segment 3 **506**, which are similar to the data segments illustrated in **Figure 4**. In this example, these data segments form a packet **508**, which is placed into packet payload **510** within data packet **512**. Additionally, data packet **512** contains CRC **514**, which is used for error checking. Additionally, routing header **516** and transport **518** are present in data packet **512**. Routing header **516** is used to identify source and destination ports for data packet **512**. Transport header **518** in this example specifies the destination queue pair for data packet **512**. Additionally, transport header **518** also provides information such as the operation code, packet sequence number, and partition for data packet **512**. The operating code identifies whether the packet is the first, last, intermediate, or only packet of a message. The operation code also specifies whether the operation is a send RDMA write, read, or atomic. The packet sequence number is initialized when communications is established and increments each time a queue pair creates a new packet. Ports of an end node may be configured to be members of one or more possibly overlapping sets called partitions.

Referring to **Figure 6**, a schematic diagram illustrating a portion of a network computing system is depicted in accordance with the present invention. The

Docket No. AUS9-2000-0628-US1

network computing system **600** in **Figure 6** includes a host processor node **602** and a host processor node **604**. Host processor node **602** includes a HCA **606** and host processor node **604** includes a HCA **608**. The network computing system **600** in **Figure 6** includes a SAN fabric **610** which includes a switch **612** and a switch **614**. The SAN fabric **610** in **Figure 6** includes a link coupling HCA **606** to switch **612**; a link coupling switch **612** to switch **614**; and a link coupling HCA **608** to switch **614**.

In the example transactions, host processor node **602** includes a client process A **616**, and host processor node **604** includes a client process B **618**. Client process A **616** interacts with HCA hardware **606** through QP **620**. Client process B **618** interacts with HCA hardware **608** through QP **622**. QP **620** and QP **622** are data structures. QP **620** includes include a send work queue **624** and a receive work queue **626**. QP **622** includes include a send work queue **628** and a receive work queue **630**.

Process A **616** initiates a message request by posting WQEs to the send queue **624** of QP **620**. Such a WQE is illustrated by WQE **428** in **Figure 4**. The message request of client process A **616** is referenced by a gather list contained in the send WQE **428**. Each data segment in the gather list points to a virtually contiguous local memory region, which contains a part of the message. This is indicated by data segments 4 **438**, 5 **440**, and 6 **442**, which respectively hold message parts 4, 5, and 6.

Hardware in HCA **606** reads the WQE and segments the message stored in virtual contiguous buffers into packets, such as packet **512** in **Figure 5**. Packets are routed through the SAN fabric **610**, and for reliable

Docket No. AUS9-2000-0628-US1

transfer services, are acknowledged by the final destination end node, which in this case is host processor node **604**. If not successively acknowledged, the packet is retransmitted by the source end node, host processor node **602**. Packets are generated by source endnodes and consumed by destination endnodes.

In reference to **Figure 7**, a schematic diagram illustrating the network addressing used in a distributed networking system is depicted in accordance with the present invention. A host name provides a logical identification for a host node, such as a host processor node or I/O adapter node. The host name identifies the endpoint for messages such that messages are destined for processes residing on an end node specified by the host name. Thus, there is one host name per node, but a node can have multiple CAs.

A single IEEE assigned 64-bit identifier (EUI-64) **702** is assigned to each component. A component can be a switch, router, or CA.

One or more globally unique ID (GUID) identifies **704** are assigned per CA port **706**. Multiple GUIDs (a.k.a. IP addresses) can be used for several reasons, some of which are provided by the following examples. In one embodiment, different IP addresses identify different partitions or services on an end node. In a different embodiment, different IP addresses are used to specify different Quality of Service (QoS) attributes. In yet another embodiment, different IP addresses identify different paths through intra-subnet routes.

One GUID **708** is assigned to a switch **710**.

A local ID (LID) refers to a short address ID used to identify a CA port within a single subnet. In one

Docket No. AUS9-2000-0628-US1

example embodiment, a subnet has up to 2^{16} end nodes, switches, and routers, and the LID is accordingly 16 bits. A source LID (SLID) and a destination LID (DLID) are the source and destination LIDs used in a local network header. A single CA port **706** has up to 2^{LMC} LIDs **712** assigned to it. The LMC represents the LID Mask Control field in the CA. A mask is a pattern of bits used to accept or reject bit patterns in another set of data.

Multiple LIDs can be used for several reasons some of which are provided by the following examples. In one embodiment, different LIDs identify different partitions or services in an end node. In another embodiment, different LIDs are used to specify different QoS attributes. In yet a further embodiment, different LIDs specify different paths through the subnet.

A single switch port **714** has one LID **716** associated with it.

A one-to-one correspondence does not necessarily exist between LIDs and GUIDs, because a CA can have more or less LIDs than GUIDs for each port. For CAs with redundant ports and redundant conductivity to multiple SAN fabrics, the CAs can, but are not required to, use the same LID and GUID on each of its ports.

Referring now to **Figure 8**, a schematic diagram illustrating the partition addressing using the LMC field is depicted in accordance with the present invention. An end node **802** can have its facilities divided into several divisions, with each division allocated to one of several Logical Partitions (LPARs) **804** and **806**. The multiple partitions **804** and **806** of the end node **802** may communicate through the same HCA or TCA port **810**.

Docket No. AUS9-2000-0628-US1

Packets are directed across the SAN and to a port **810** via the LID address in the SAN packet. The problem arises when packets destined for different LPARS, **804** or **806**, must pass through the same CA port **810**. The present invention solves this problem by using the LMC to create more specific routing directions within the end node and port addresses within a packet.

Multiple LIDs can be assigned to a port by using the LMC capability of the SAN packet. The LMC can be any number of bits, but in one embodiment there are three bits. In this embodiment, up to eight lower order bits (2^{LMC} , $LMC = 3$) of the LID can be designated as addresses that get targeted to a particular port. The lower order bits designated by the LMC act as an addresses within an address. Whereas the LID designates the port **810** to which a packet is sent, the lower order bits designate the specific LPAR **804** or **806** within the end node **802** to which a packet is to be routed.

By using the LMC to create partition routing instructions, the present invention provides an easy and efficient routing mechanism within a single end node without the need for additional hardware or processing resources.

It is important to note that while the present invention has been described in the context of a fully functioning data processing system, those of ordinary skill in the art will appreciate that the processes of the present invention are capable of being distributed in the form of a computer readable medium of instructions and a variety of forms and that the present invention applies equally regardless of the particular type of signal bearing media actually used to carry out the

Docket No. AUS9-2000-0628-US1

distribution. Examples of computer readable media include recordable-type media, such as a floppy disk, a hard disk drive, a RAM, CD-ROMs, DVD-ROMs, and transmission-type media, such as digital and analog communications links, wired or wireless communications links using transmission forms, such as, for example, radio frequency and light wave transmissions. The computer readable media may take the form of coded formats that are decoded for actual use in a particular data processing system.

The description of the present invention has been presented for purposes of illustration and description, and is not intended to be exhaustive or limited to the invention in the form disclosed. Many modifications and variations will be apparent to those of ordinary skill in the art. The embodiment was chosen and described in order to best explain the principles of the invention, the practical application, and to enable others of ordinary skill in the art to understand the invention for various embodiments with various modifications as are suited to the particular use contemplated.

CLAIMS:

What is claimed is:

1. A method for routing data packets to multiple partitions within a single end node, comprising:
 assigning a range of local identification addresses (LIDs) to a channel adapter port in an end node; and
 assigning bits within the local identification addresses to specify which of several partitions within the end node is being addressed.
2. The method of claim 1, wherein the bits are lower order bits.
3. The method according to claim 1, wherein the channel adapter port is connected to a system area network.
4. The method according to claim 1, wherein:
 the network contains two raised to the N power end nodes, switches, and routers; and
 the number of bits in a local identification address equals N.
5. The method according to claim 2, wherein the lower order bits assigned to partitions are designated by a local identification mask control (LMC) field.
6. The method according to claim 5, wherein the local identification mask control can be any number of bits.
7. The method according to claim 5, wherein a number of

Docket No. AUS9-2000-0628-US1

lower order bits assigned to addressing within a port is up to two raised to the local identification mask control power.

8. The method according to claim 7, wherein the different local identification addresses of a port identify different partitions within the end node.

9. A computer program product in a computer readable medium for use in a data process system for routing data packets to multiple partitions within a single end node, the computer program product comprising:

instructions for assigning a local identification address to a channel adapter port in an end node; and

instructions for assigning bits within the local identification address to a specific partition within the end node.

10. The computer program product of claim 9, wherein the bits are lower order bits.

11. The computer program product according to claim 9, further comprising instructions for connecting the channel adapter port to a system area network.

12. The computer program product according to claim 9, wherein:

if the network contains two raised to the N power end nodes, switches, and routers;

the number of bits in a local identification address equals N.

Docket No. AUS9-2000-0628-US1

13. The computer program product according to claim 10, wherein the lower order bits assigned to partitions are designated by a local identification mask control (LMC) field.

14. The computer program product according to claim 13, wherein the local identification mask control can be any number of bits.

15. The computer program product according to claim 13, wherein the number of lower order bits assigned to addressing within a port is up to two raised to the local identification mask control power.

16. The computer program product according to claim 15, wherein the different local identification addresses of a port identify different partitions within the end node.

17. A system for routing data packets to multiple partitions within a single end node, comprising:

means for assigning a local identification address (LID) to a channel adapter port in a network end node; and

means for assigning lower order bits within the local identification addresses to a specific partition within the end node.

18. The system according to claim 17, wherein the channel adapter port is connected to a system area network.

19. The method according to claim 17, wherein:

Docket No. AUS9-2000-0628-US1

the network contains two raised to the N power end nodes, switches, and routers; and

the number of bits in a local identification address equals N.

20. The system according to claim 17, wherein the lower order bits assigned to partitions are designated by a local identification mask control (LMC) field.

21. The method according to claim 20, wherein the LMC can be any number of bits.

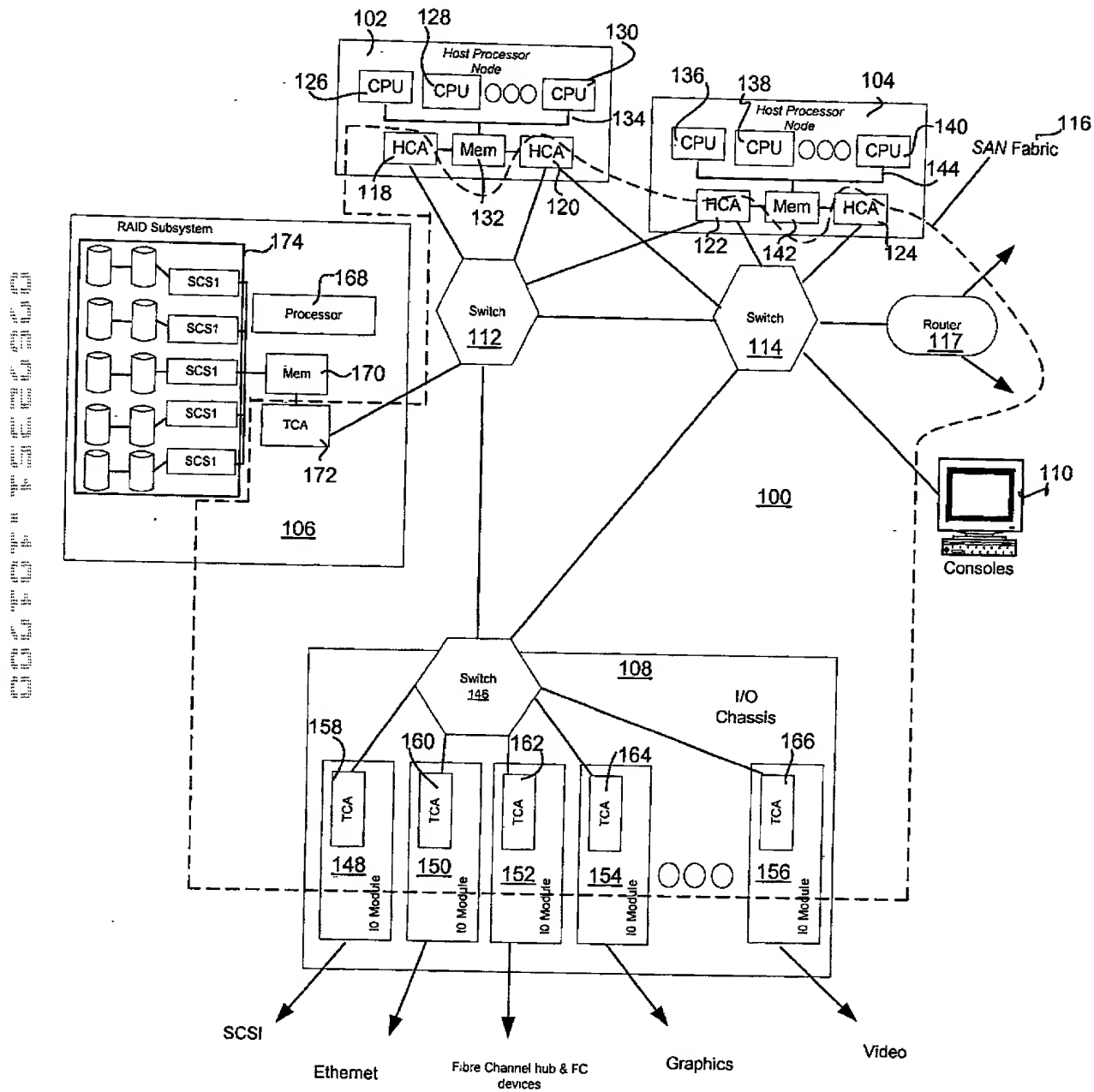
22. The system according to claim 20, wherein the number of lower order bits assign to addressing within a port is up to two raised to the local identification mask control power.

23. The system according to claim 22, wherein the different local identifier addresses of a port identify different partitions within the end node.

ABSTRACT OF THE DISCLOSURE**END NODE PARTITIONING USING LMC FOR A SYSTEM AREA NETWORK**

A method for routing System Area Network (SAN) packets to multiple partitions within a single end node is provided. A range of Local Identification addresses (LIDs) are assigned to a channel adapter port within the SAN. Lower order bits within the LID are then assigned to select the particular partition in the end node. The Local ID Mask Control (LMC) field is used to assign multiple LIDs to a single port, using those low order bits to then route the message to the appropriate partition in the end node.

Figure 1

AUS9-2000-0628-US1
Page 1 of 8

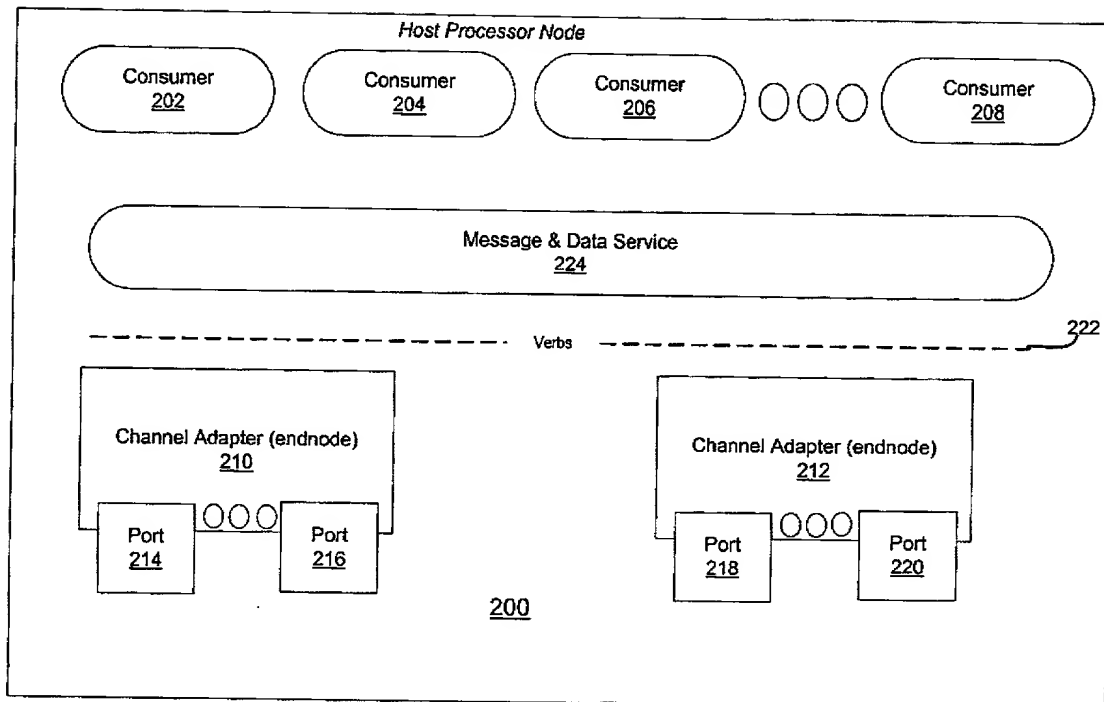


Figure 2

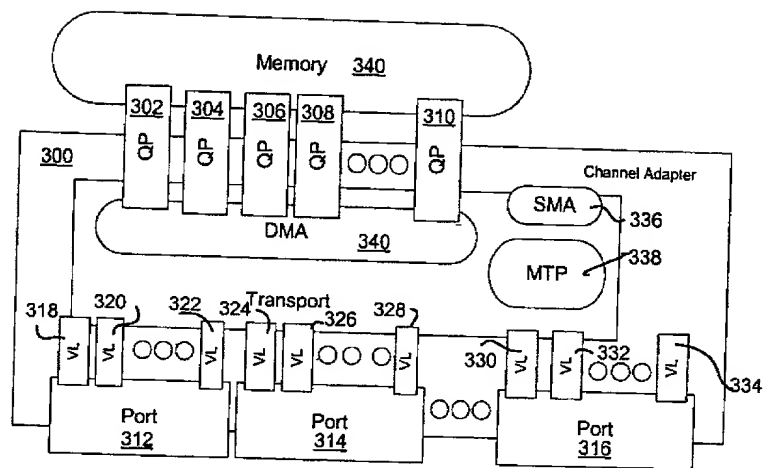


Figure 3

AUS9-2000-0628-US1
Page 3 of 8

Figure 4

AUS9-2000-0628-US1
Page 4 of 8

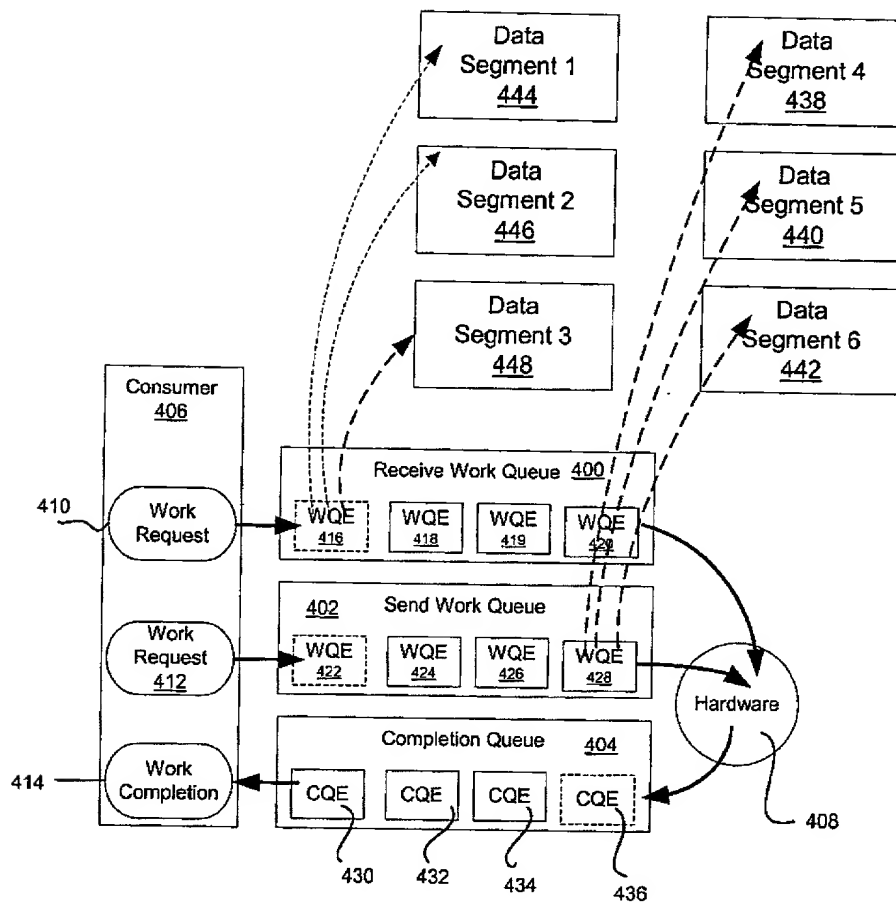
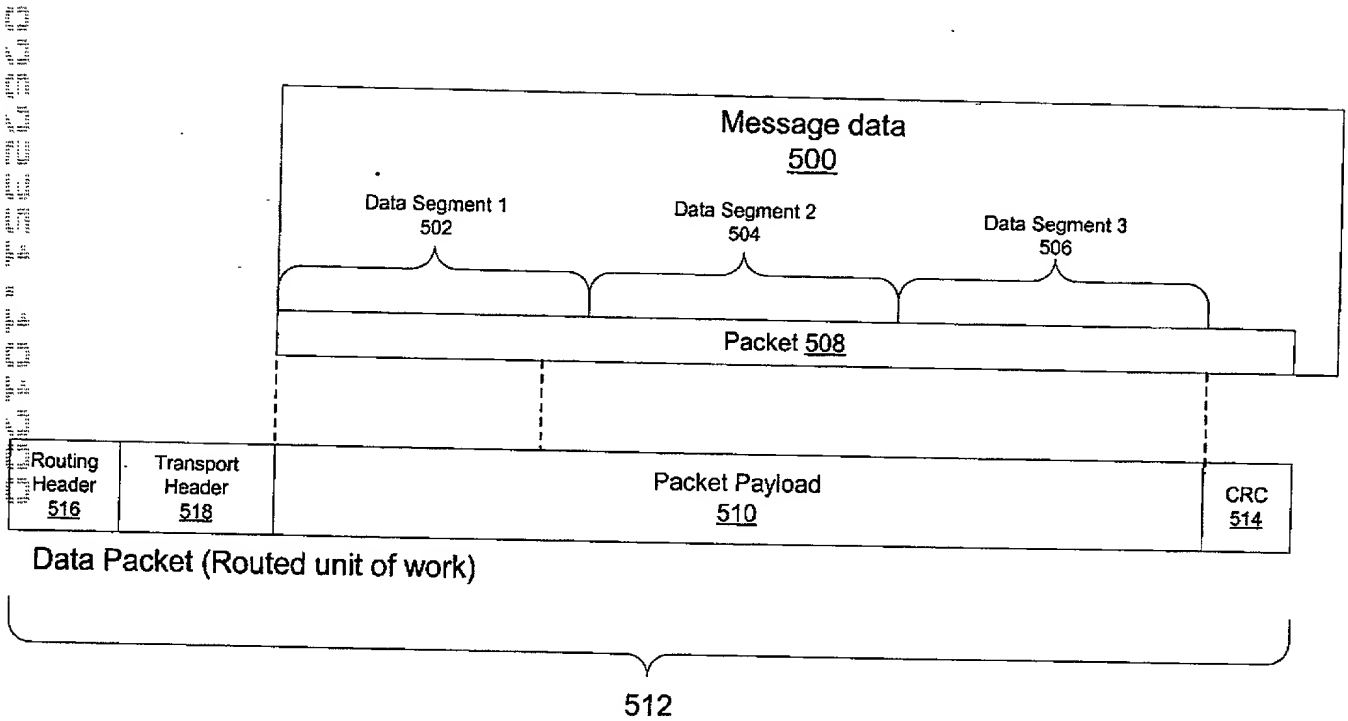


Figure 5

AUS9-2000-0628-US1
Page 5 of 8



AUS9-2000-0628-US1
Page 6 of 8

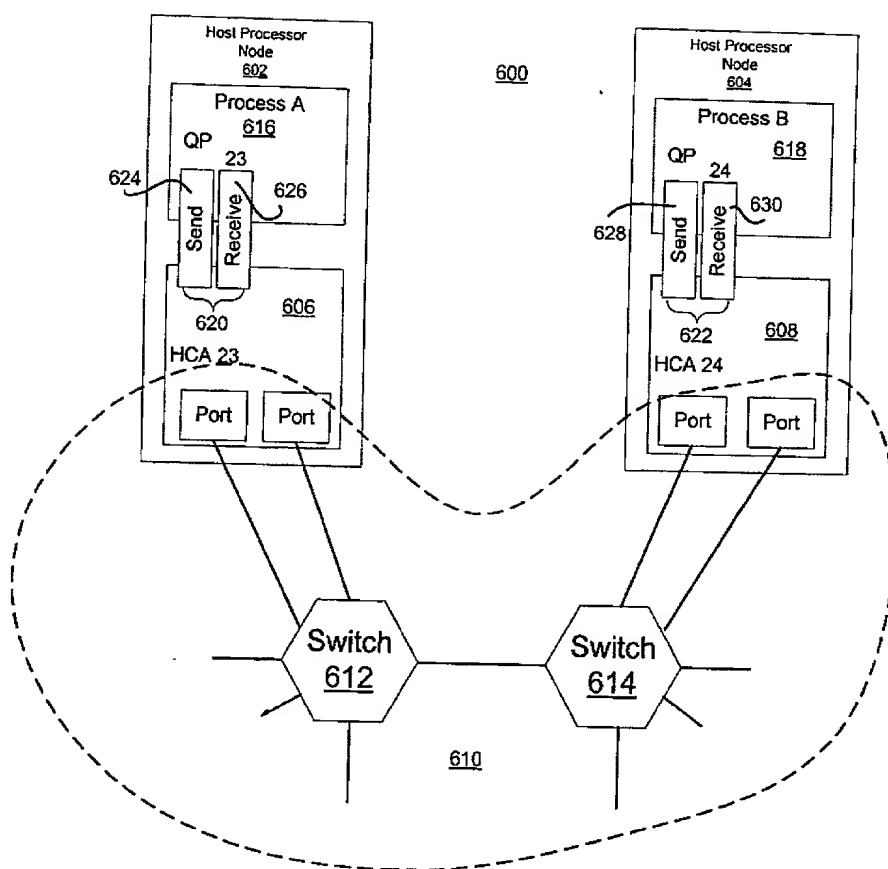
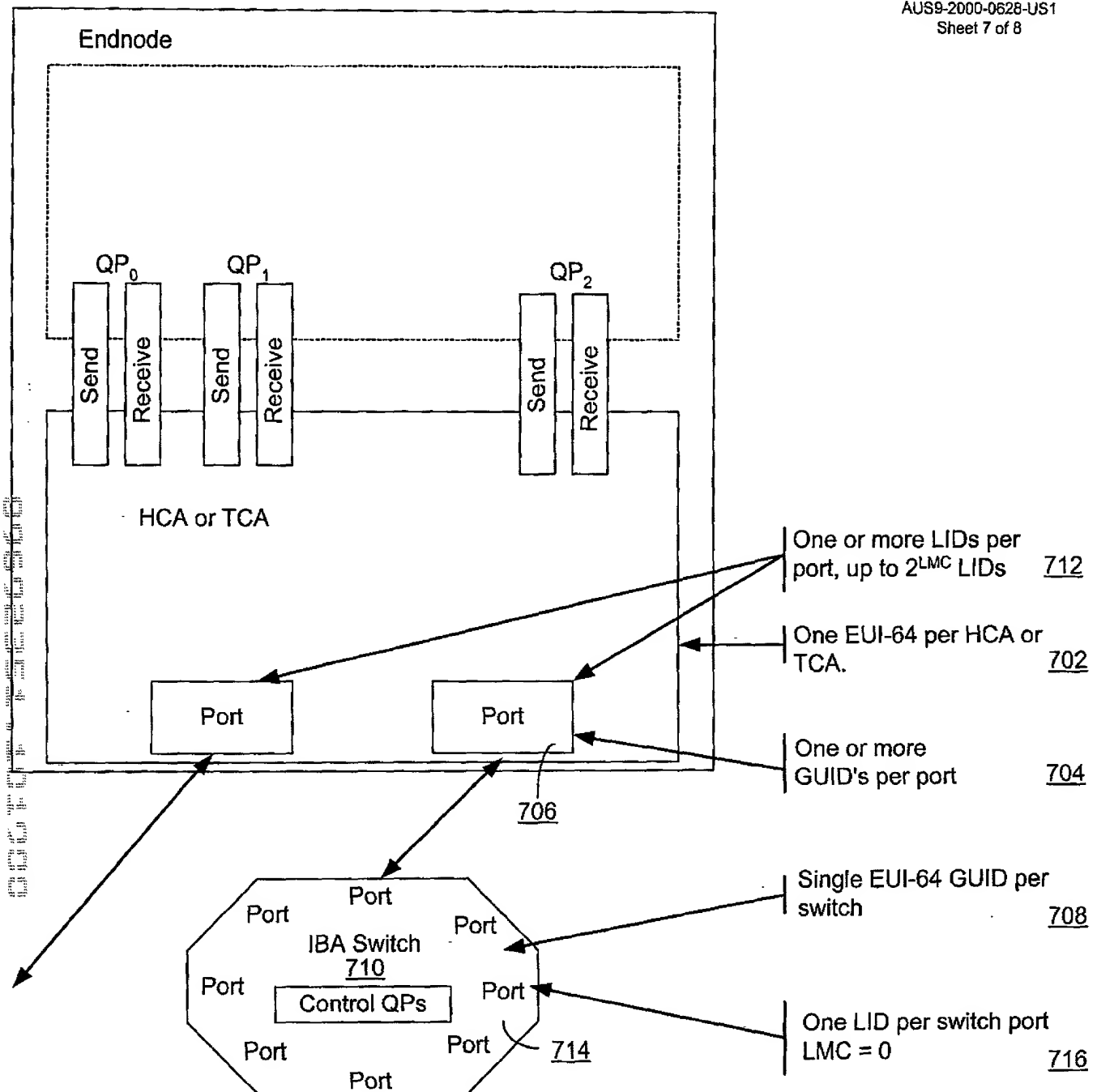
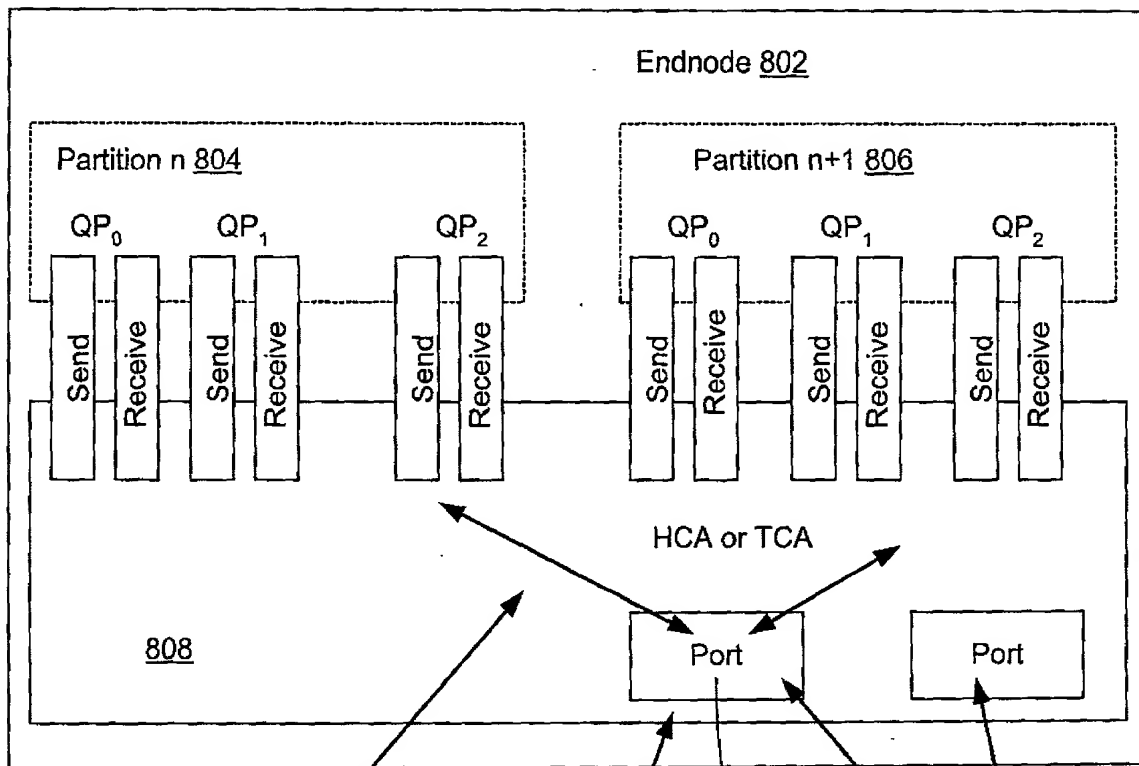


Figure 7

AUS9-2000-0628-US1
Sheet 7 of 8





Packet routing to correct Partition based on low-order bits of the LID

One or more LIDs per port, up to 2^{LMC} LIDs

To SAN Network

Figure 8

AUS9-2000-0628-US1
Sheet 8 of 8

**DECLARATION AND POWER OF ATTORNEY FOR
PATENT APPLICATION**

As a below named inventor, I hereby declare that:

My residence, post office address and citizenship are as stated below next to my name;

I believe I am the original, first and sole inventor (if only one name is listed below) or an original, first and joint inventor (if plural names are listed below) of the subject matter which is claimed and for which a patent is sought on the invention entitled

END NODE PARTITIONING USING LMC FOR A SYSTEM AREA NETWORK

the specification of which (check one)

X is attached hereto.

___ was filed on _____
as Application Serial No. _____
and was amended on _____
(if applicable)

I hereby state that I have reviewed and understand the contents of the above identified specification, including the claims, as amended by any amendment referred to above.

I acknowledge the duty to disclose information which is material to the patentability of this application in accordance with Title 37, Code of Federal Regulations, §1.56.

I hereby claim foreign priority benefits under Title 35, United States Code, §119 of any foreign application(s) for patent or inventor's certificate listed below and have also identified below any foreign application for patent or inventor's certificate having a filing date before that of the application on which priority is claimed:

Prior Foreign Application(s):	Priority Claimed
_____	___ Yes___ No
(Number)	(Country) (Day/Month/Year)

I hereby claim the benefit under Title 35, United States Code, §120 of any United States application(s) listed below and, insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States application in the manner provided by the first paragraph of Title 35, United States Code, §112, I acknowledge the duty to disclose information material to the patentability of this application as defined in Title 37, Code of Federal Regulations, §1.56 which occurred between the filing date of the prior application and the national or PCT international filing date of this application:

_____	_____	_____
(Application Serial #)	(Filing Date)	(Status)

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be

true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

POWER OF ATTORNEY: As a named inventor, I hereby appoint the following attorneys and/or agents to prosecute this application and transact all business in the Patent and Trademark Office connected therewith.

John W. Henderson, Jr., Reg. No. 26,907; Thomas E. Tyson, Reg. No. 28,543; James H. Barksdale, Jr., Reg. No. 24,091; Casimer K. Salys, Reg. No. 28,900; Robert M. Carwell, Reg. No. 28,499; Douglas H. Lefevre, Reg. No. 26,193; Jeffrey S. LaBaw, Reg. No. 31,633; David A. Mims, Jr., Reg. 32,708; Volel Emile, Reg. No. 39,969; Anthony V. England, Reg. No. 35,129; Leslie A. Van Leeuwen, Reg. No. 42,196; Christopher A. Hughes, Reg. No. 26,914; Edward A. Pennington, Reg. No. 32,588; John E. Hoel, Reg. No. 26,279; Joseph C. Redmond, Jr., Reg. No. 18,753; Marilyn S. Dawkins, Reg. No. 31,140; Mark E. McBurney, Reg. No. 33,114; Duke W. Yee, Reg. No. 34,285; Colin P. Cahoon, Reg. No. 38,836; Stephen R. Loe, Reg. No. 43,757; Stephen J. Walder, Jr., Reg. No. 41,534; Charles D. Stepps, Jr., Reg. No. 45,880; Stephen R. Tkacs, Reg. No. 46,430, and Christopher P. O'Hagan, Reg. No. P-46,966, Lisa L.B. Yociss, Reg. No. 36,975.

Send correspondence to: Duke W. Yee, Carstens, Yee & Cahoon, LLP, P.O. Box 802334, Dallas, Texas 75380 and direct all telephone calls to Duke W. Yee, (972) 367-2001

FULL NAME OF SOLE OR FIRST INVENTOR: Thomas Anthony Gregg

INVENTORS SIGNATURE: _____ DATE: _____

RESIDENCE: 121 Bellevue Road
Highland, New York 12528

CITIZENSHIP: United States

POST OFFICE ADDRESS: SAME AS ABOVE

FULL NAME OF SECOND INVENTOR: Gregory Francis Pfister

INVENTORS SIGNATURE: Gregory Francis Pfister DATE: 10/13/00

RESIDENCE: 5905 Sir Ivor Cove
Austin, Texas 78746

CITIZENSHIP: United States

POST OFFICE ADDRESS: SAME AS ABOVE

FULL NAME OF THIRD INVENTOR: Renato John Recio

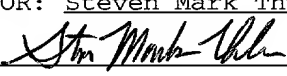
INVENTORS SIGNATURE:  DATE: 10/16/2000

RESIDENCE: 6707 Winnipeg Cove
Austin, Texas 78759

CITIZENSHIP: United States

POST OFFICE ADDRESS: SAME AS ABOVE

FULL NAME OF FOURTH INVENTOR: Steven Mark Thurber

INVENTORS SIGNATURE:  DATE: 10/12/2000

RESIDENCE: 8308 Ephraim Road
Austin, Texas 78717

CITIZENSHIP: United States

POST OFFICE ADDRESS: SAME AS ABOVE

Docket No. AUS9-2000-0628-US1

**DECLARATION AND POWER OF ATTORNEY FOR
PATENT APPLICATION**

As a below named inventor, I hereby declare that:

My residence, post office address and citizenship are as stated below next to my name;

I believe I am the original, first and sole inventor (if only one name is listed below) or an original, first and joint inventor (if plural names are listed below) of the subject matter which is claimed and for which a patent is sought on the invention entitled

END NODE PARTITIONING USING LMC FOR A SYSTEM AREA NETWORK

the specification of which (check one)

X is attached hereto.

— was filed on _____
as Application Serial No. _____
and was amended on _____
(if applicable)

I hereby state that I have reviewed and understand the contents of the above identified specification, including the claims, as amended by any amendment referred to above.

I acknowledge the duty to disclose information which is material to the patentability of this application in accordance with Title 37, Code of Federal Regulations, §1.56.

I hereby claim foreign priority benefits under Title 35, United States Code, §119 of any foreign application(s) for patent or inventor's certificate listed below and have also identified below any foreign application for patent or inventor's certificate having a filing date before that of the application on which priority is claimed:

Prior Foreign Application(s):

Priority Claimed

_____ (Number)	_____ (Country)	_____ (Day/Month/Year)	____ Yes ____ No
-------------------	--------------------	---------------------------	------------------

I hereby claim the benefit under Title 35, United States Code, §120 of any United States application(s) listed below and, insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States application in the manner provided by the first paragraph of Title 35, United States Code, §112, I acknowledge the duty to disclose information material to the patentability of this application as defined in Title 37, Code of Federal Regulations, §1.56 which occurred between the filing date of the prior application and the national or PCT international filing date of this application:

_____ (Application Serial #)	_____ (Filing Date)	_____ (Status)
---------------------------------	------------------------	-------------------

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be

OCT 13 '00 10:03 FR AUSTIN IPL IMD

5128383516 TO 72929400

P.06/15

Docket No. AUS9-2000-0628-US1

true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

POWER OF ATTORNEY: As a named inventor, I hereby appoint the following attorneys and/or agents to prosecute this application and transact all business in the Patent and Trademark Office connected therewith.

John W. Henderson, Jr., Reg. No. 26,907; Thomas E. Tyson, Reg. No. 28,543; James H. Barksdale, Jr., Reg. No. 24,091; Casimer K. Salys, Reg. No. 28,900; Robert M. Carwell, Reg. No. 28,499; Douglas H. Lefevre, Reg. No. 26,193; Jeffrey S. LaBaw, Reg. No. 31,633; David A. Mims, Jr., Reg. 32,708; Volel Emile, Reg. No. 39,969; Anthony V. England, Reg. No. 35,129; Leslie A. Van Leeuwen, Reg. No. 42,196; Christopher A. Hughes, Reg. No. 26,914; Edward A. Pennington, Reg. No. 32,588; John E. Hoel, Reg. No. 26,279; Joseph C. Redmond, Jr., Reg. No. 18,753; Marilyn S. Dawkins, Reg. No. 31,140; Mark E. McBurney, Reg. No. 33,114; Duke W. Yee, Reg. No. 34,285; Colin P. Cahoon, Reg. No. 38,836; Stephen R. Loe, Reg. No. 43,757; Stephen J. Walder, Jr., Reg. No. 41,534; Charles D. Stepps, Jr., Reg. No. 45,880; Stephen R. Tkacs, Reg. No. 46,430, and Christopher P. O'Hagan, Reg. No. P-46,966, Lisa L.B. Yociss, Reg. No. 36,975.

Send correspondence to: Duke W. Yee, Carstens, Yee & Cahoon, LLP, P.O. Box 802334, Dallas, Texas 75380 and direct all telephone calls to Duke W. Yee, (972) 367-2001

FULL NAME OF SOLE OR FIRST INVENTOR: Thomas Anthony Green

INVENTORS SIGNATURE: Thomas Anthony Green DATE: October 13, 2000

RESIDENCE: 121 Bellevue Road
Highland, New York 12528

CITIZENSHIP: United States

POST OFFICE ADDRESS: SAME AS ABOVE

FULL NAME OF SECOND INVENTOR: Gregory Francis Pfister

INVENTORS SIGNATURE: _____ DATE: _____

RESIDENCE: 5905 Sir Ivor Cove
Austin, Texas 78746

CITIZENSHIP: United States

POST OFFICE ADDRESS: SAME AS ABOVE

OCT 13 '00 10:03 FR AUSTIN IPL IMD

5128383516 TO 72929400

P.07/15

Docket No. AUS9-2000-0628-US1

FULL NAME OF THIRD INVENTOR: Renato John Rocio

INVENTORS SIGNATURE: _____ DATE: _____

RESIDENCE: 6707 Winnipeg Cove
Austin, Texas 78759CITIZENSHIP: United StatesPOST OFFICE ADDRESS: SAME AS ABOVEFULL NAME OF FOURTH INVENTOR: Steven Mark Thurber

INVENTORS SIGNATURE: _____ DATE: _____

RESIDENCE: 8308 Ephraim Road
Austin, Texas 78717CITIZENSHIP: United StatesPOST OFFICE ADDRESS: SAME AS ABOVE